# Computational Intelligence Lab: Road Segmentation

Thomas Cambier, François Clément, Justin Dallant, Thibault Dardinier
Department of Computer Science, ETH Zurich, Switzerland

***Abstract*—Road detection from aerial images is a challenging problem nowadays in a wide variety of sectors such as road mapping, city planning and traffic management. The ever growing amount of data available, as well as more precise images, have led to learning algorithms, and in particular Convolutional Neural Networks (CNNs) to be particularly adapted to this problem. In this project, we used the most recent CNN methods combined with morphological models, data and test-time augmentation techniques to tackle in the best possible way the road segmentation task.**

## I. Introduction

The goal of road segmentation is to segment an input aerial image into two categories: road and non-road (or background). To this extent, each one of the pixels constituting the image is attributed a label telling whether it belongs to a road or not and we aim at training a model that can achieve high accuracy in determining correctly these labels.

Our approach makes use of a Convolutional Neural Network (CNN) after a pre-processing step in which features are added to the data in order to increase the accuracy of the model. Furthermore, we implemented several baselines to compare our solution to.

In Section II we go over some related work on the subject of road segmentation, while in Section III are presented the baselines we implemented as well as our new approach. Section IV contains an overview of the results achieved in this paper before concluding in Section V.

## II. Related Work

We will give here a quick overview on the different methods available for road segmentation as well as a quick description of CNNs. [1] provides a good starting point to survey the different methods as well as their drawbacks. All semi-automatic methods such as the mean shift algorithm or some active contour models (snakes for example) are not adapted since we cannot provide the required input for each image. The dynamic programming approaches also suffer from a required parameter which does not quite fit with our problem (or would add quite a lot of work whereas the Neural network approach fits very well). The morphological approaches are interesting and fit very well with the relatively "simple" shapes of roads but have a number of disadvantages such as depending on the choice of road structure elements or generally low accuracy. However, this method can combine very well with other methods, for example as post-processing following a CNN.

The most interesting way to tackle the road segmentation problem given the data and sample images given is to use a Convolutional Neural Network. They have been extensively used in image computing since the late 1980's thanks to a number of advantages, for example translation invariance (wherever the road is in the image, we want to be able to identify it as a road). Training the CNN is feasible given that we have a large data set of classified images but even with the data available it may suffer from overfitting and more images need to be generated (or diversifying the original set, for example by rotating the images). To compensate for the lack of precision of a CNN (non-road areas are often selected), it is interesting to then use a morphological model (for example to eliminate parking zones which are not "road-shaped").

## III. Models and Methods

### A. Baselines

There are ten different baselines based on two input classes, "base" and "morpho". The "base" input consists of the base pixels in the RGB (Red-Green-Blue) color model, thus 3-dimensional and the "morpho" input consists of the base pixels in the HSV (Hue-Saturation-Lightness) color model as well as four features added during a pre-processing phase, thus 7-dimensional.

To obtain these features, we segment the image using the Felzenszwalb method [2] which is a graph-based image segmentation method. At the beginning of the computation, the input image is represented as a graph containing edges between every adjacent pixels with a weight corresponding to the similarity of those pixels. Then, edges of the minimum spanning tree of that graph are processed in order of increasing weight and contracted whenever the weight of the edge is considered small compared to the weights of edges that were present in the two components it connects. This method was preferred over other segmentation methods such as K-means because it showed superior computational efficiency in this context. After the segmentation, each segment is smoothed and its holes are filled using topological closure operations so that we can compute its medial axis or skeleton. "The medial axis of an object is the set of all points having more than one closest point on the objects boundary. It is often called the topological skeleton, because it is a 1-pixel wide skeleton of the object, with the same connectivity as the original object." [3] Then, we trim the skeleton to get rid of

small spurs that are simply the result of noise in the segment shape.

Finally, these skeletons allow us to compute the following features which are added to all pixels from a segment:

- Average width of segment (average distance from the skeleton to the closest point on the border of the segment)
- Standard deviation to the width
- Elongation (length of skeleton squared divided by the area of the segment)
- Number of endpoints of the skeleton

Some other features to quantify the "complexity" of the shape were tried such as the ratio between the length of the real skeleton and the length of the skeleton of a very smoothed out version of the segment, the ratio between the length of the skeleton and the distance between the two most far apart points of the skeleton or the mean squared distance from the skeleton to the straight line segment joining the two most far apart points of the skeleton. They were abandoned because none of these features appeared to be very robust and varied greatly between very similar shapes due to noise on the boundary that changed the skeleton.

From either of these two input classes, we form new ones by adding to each pixel either its eight neighbours (base_n, 27-dimensional; morpho_n, 63-dimensional), its distance to the road mean point (base_d, 4-dimensional; morpho_d, 8-dimensional) or both, adding first the distance and, then, the neighbours (base_dn, 36-dimensional; morpho_dn, 72-dimensional) [4]. The distance is computed by taking the means of all pixels labeled as roads over all images from the training set.

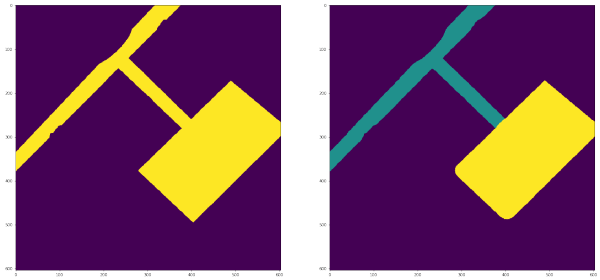
Finally, the last two baselines are obtained by cutting the input image in patches of 8x8 pixels for both input classes and measuring for each original feature its mean and standard deviation, thus doubling the number of features (base_p, 6-dimensional; morpho_p, 14-dimensional) and dividing by 64 the number of patches considered in the image.

The neural networks used to train the different baselines all consist of two hidden layers with manually-tailored sizes and sigmoid activation functions. The final numbers of neurons were obtained through experimentation which, though not optimal, led to satisfying results.

### *B. The Convolutional Neural Network*

The new approach that we tried uses a CNN which classifies patches of 8x8 pixels, based on a context of 80x80 pixels. It uses a sequence of 3x3 convolutional filters with 2x2 maximum pooling. The layer properties are described in Table I.

Dropout is used to minimize overfitting and we use exponential linear units (ELU) as activation functions to avoid dead neurons as well as to provide faster training and higher accuracy [5]. We have a total of 1,208,002 parameters, all of which are trainable. To train them, we use Keras



**(a)** Original segment **(b)** Reconstruction

Figure 1. Example of the reconstruction of a segment with a road and the roof of a building.

with a Tensor flow backend on 200 epochs, with 100x512 parameters per epoch. On a Tesla K80 GPU provided by the Google Collab service, this takes approximately 4 hours. In addition we use two types of augmentation: data and test time augmentation. For the data augmentation, we extract random 80x80 patches which we rotate, mirror or colour-shift (mirroring is used near the borders). For the time test augmentation, we replace the regular classification of a patch by the classification of different averages and we average the results to obtain the result for the initial patch. The different variations possible are small zooms, small shears, mirroring, rotations and small shifts.

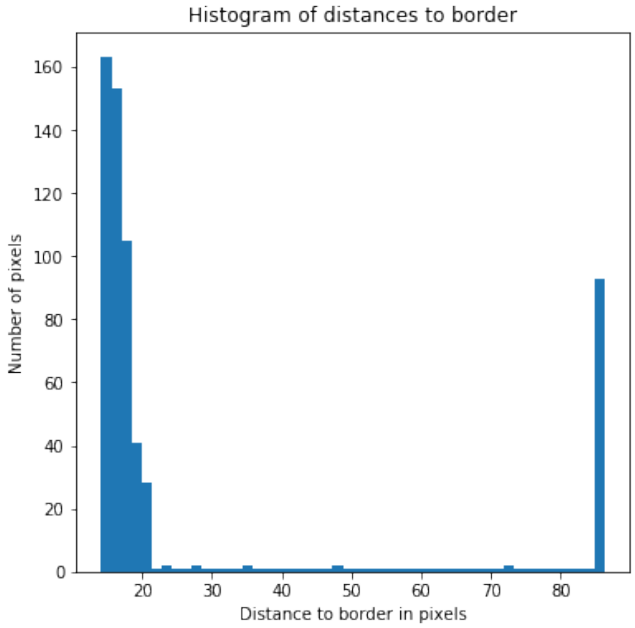



Figure 2. Histogram of distances between skeleton and border.

### *C. An attempted post-processing*

The goal of this approach was to use a similar method to the pre-processing to classify the segments into road/non-road using morphological features, then use the average

| Layer | Type | Output Shape | Number of Parameters |
|---|---|---|---|
| conv2d_12 | Conv2d | (None, 80, 80, 32) | 896 |
| max_pooling2d_12 | MaxPooling | (None, 40, 40, 32) | 0 |
| elu_15 | ELU | (None, 40, 40, 32) | 0 |
| dropout_15 | Dropout | (None, 40, 40, 32) | 0 |
| conv2d_13 | Conv2d | (None, 40 ,40 ,64) | 18496 |
| max_pooling2d_13 | MaxPooling | (None, 20, 20, 64) | 0 |
| elu_16 | ELU | (None, 20, 20, 64) | 0 |
| dropout_16 | Dropout | (None, 20, 20, 64) | 0 |
| conv2d_14 | Conv2d | (None, 20, 20, 128) | 73856 |
| max_pooling2d_14 | MaxPooling | (None, 10, 10, 128) | 0 |
| elu_17 | ELU | (None, 10 ,10 ,128) | 0 |
| dropout_17 | Dropout | (None, 10, 10, 128) | 0 |
| conv2d_15 | Conv2d | (None, 10, 10, 256) | 295168 |
| max_pooling2d_15 | MaxPooling | (None, 5, 5, 256) | 0 |
| elu_18 | ELU | (None, 5, 5, 256) | 0 |
| dropout_18 | Dropout | (None, 5, 5, 256) | 0 |
| flatten_3 | Flatten | (None, 6400) | 0 |
| dense_6 | Dense | (None, 128) | 819328 |
| elu_19 | ELU | (None, 128) | 0 |
| dropout_19 | Dropout | (None, 128) | 0 |
| dense_7 | Dense | (None, 2) | 256 |

Table I
LAYERS OF THE CNN

classification over the segment given by the CNN, all of this to combine the local nature of our patch-based CNN with more global elements. The main difference with the pre-processing is that for the fourth channel of the Felzenszwalb segmentation we use the mask produced by the CNN. However, this method leads to two problems.

The first one appears on segments like the one represented on Image 1. To tackle this problem we used the following method. We also use the observation that when road and non-road parts accidentally end up in the same segment, we can often tell them apart visually. We introduce here the notion of minimal skeleton: we take the distance transform of the segment (the distance from each pixel in the segment to the border), we then take the skeleton and only keep the edges which connect two local maxima of the distance transform to obtain the minimal skeleton [6]. We only compute an approximation of this skeleton in our algorithm due to time and computation cost issues. This skeleton is less sensitive to noise while preserving a lot of the morphological information.

With this minimal skeleton, we take the list of distances from each point to the border of the skeleton and we can make an histogram out of this (Image 2). We can then interpret these points as a Gaussian distribution, estimated using Kernel Density Estimation, giving us a smoother version of the histogram (Image 3). We then observe bumps corresponding to different parts of the segment (road and rooftop for example). The local minima of the distribution then give us the points where we need to break the segment (Image 5). Once this has-been done, we go back to the regular skeleton to reconstruct the different segments (Image 1).

The second problem which came up was that to train a model based on these features, we need the output of both

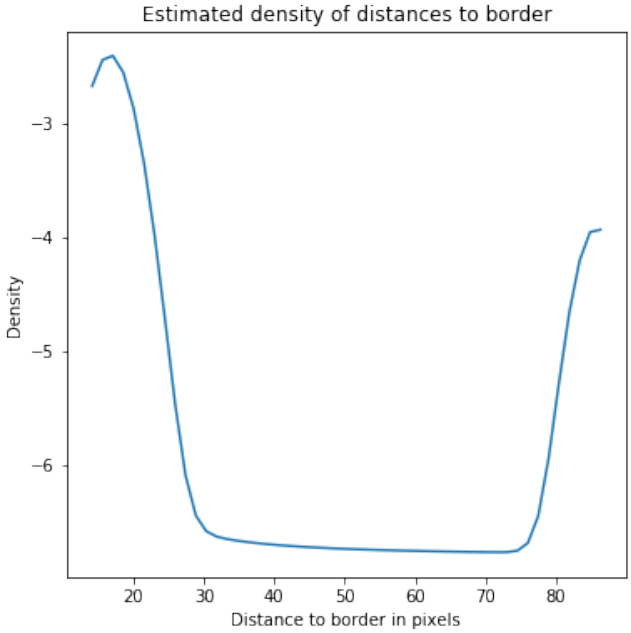
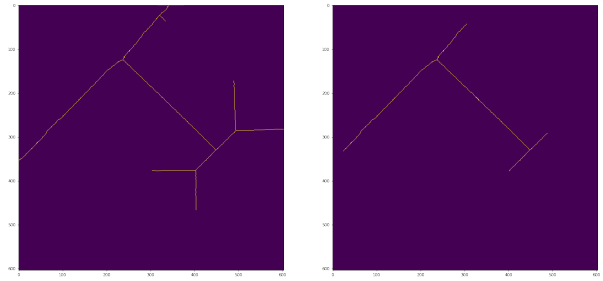



Figure 3. Kernel Density Estimation.

the first model and the groundtruth labelling. This means that both models cannot be trained with the same data set. However, even with data augmentation, we found we were lacking in data. This was the real blocking point which led to the method being abandoned.

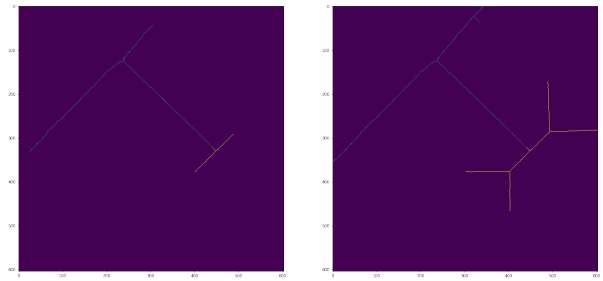
## IV. RESULTS

To train our model, we used 5-fold cross-validation. The 100 images from the training set are shuffled and divided



**(a)** Skeleton **(b)** Minimal skeleton

Figure 4. Original skeleton and its minimal version from the example segment.



**(a)** Min broken skeleton **(b)** Broken skeleton

Figure 5. Broken versions of the minimal and original skeleton from the example segment

into five subsets of 20 images each. Then, for each one of these subsets, the data augmentation (e.g. computation of the mean road point) and the training are done over the 80 remaining images and the predictions are made over the subset.

The results we obtained are summarized in Table II. On the left are listed all the methods used with, on top, the four methods from our new approach described in Section III-B and, below, the ten baselines described in Section III-A. The accuracy measures the number of pixels (or patches) correctly classified over the number of total pixels, RMSE stands for Root Mean Square Error, RCC for Road Correctly Classified (number of correctly classified road pixels over total number of road pixels) and BCC for Background Correctly Classified (number of non-road pixels correctly classified over total number of non-road pixels). The best CNN method achieves an accuracy of 0.907 whereas the best baseline achieves 0.846. The CNN methods work better because all the baselines suffer from being too local: they only consider a unique pixel (or, at best, a few neighbours) and lack more global information and context. For the same reason, "morpho" and neighbours-looking methods achieve better results than their "base" counterparts.

| Name | Accuracy | RMSE | RCC | BCC |
|---|---|---|---|---|
| cnn_morpho_post | 0.907 | 0.284 | 0.592 | 0.985 |
| cnn_morpho_tta_post | 0.907 | 0.284 | 0.592 | 0.985 |
| cnn_morpho_tta | 0.904 | 0.298 | 0.805 | 0.929 |
| cnn_morpho | 0.904 | 0.298 | 0.805 | 0.929 |
| morpho_n | 0.846 | 0.312 | 0.371 | 0.965 |
| morpho_dn | 0.841 | 0.316 | 0.358 | 0.962 |
| morpho | 0.84 | 0.319 | 0.348 | 0.963 |
| morpho_d | 0.839 | 0.321 | 0.329 | 0.967 |
| base_n | 0.824 | 0.327 | 0.277 | 0.961 |
| base_dn | 0.823 | 0.328 | 0.272 | 0.962 |
| morpho_p | 0.811 | 0.317 | 0.179 | 0.966 |
| base_p | 0.803 | 0.344 | 0 | 1 |
| base_d | 0.8 | 0.356 | 0 | 1 |
| base | 0.799 | 0.357 | 0 | 1 |

Table II
RESULTS TABLE SORTED BY ACCURACY

For the three least performing baselines, RCC is set to 0 and BCC to 1 which means that all pixels are considered to be background pixels and that such a choice already achieves a quite high accuracy of 80%.

## V. CONCLUSION

In this paper, we have discussed how to tackle the problem of detecting roads from aerial images using a new approach combining a pre-processing step and a CNN. With data and test-time augmentation, this method achieved high accuracy compared to the baselines implemented. We would have liked to add a post-processing component to it but the training became highly data-consuming which led us to leave this for future work.

## REFERENCES


[1] Wang, "A review of road extraction from remote sensing images," 2016.

[2] Felzenszwalb and Huttenlocher, "Efficient graph-based image segmentation," 2004.

[3] "Medial axis computation," https://scikit-image.org/docs/0.10.x/auto_examples/plot_medial_transform.html.

[4] Zoej and Mokhtarzade, "Road detection from high resolution satellite images using artificial neural networks," 2007.

[5] Clevert, "Fast and accurate deep network learning by exponential linear units (elus)," 2015.

[6] Talbot and Terol, "Binary image segmentation using weighted skeletons," 1992.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

## Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

---

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

**Title of work** (in block letters):

COMPUTATIONAL INTELLIGENCE LAB PROJECT

**Authored by** (in block letters):
*For papers written by groups the names of all authors are required.*

| **Name(s):** | **First name(s):** |
|---|---|
| CAMBIER | THOMAS |
| CLEMENT | FRANCOIS |
| DALLANT | JUSTIN |
| DARDINIER | THIBAULT |

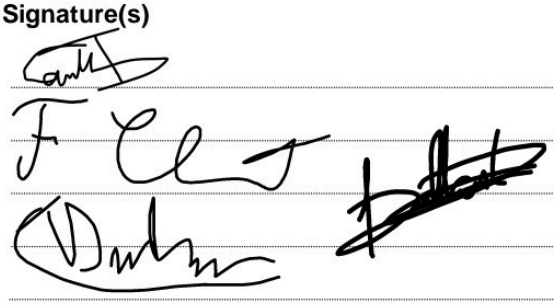
With my signature I confirm that
- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

**Place, date**

Zürich, 05/07/2019

**Signature(s)**



*For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.*